# Оптимизация вычислений нейронных сетей на графических процессорах для мобильных устройств

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА

Оплачко Николай Алексеевич
420 группа

**Научный руководитель:**
д.ф.-м.н., профессор, академик РАН
Аветисян Арутюн Ишханович

# Введение (1)

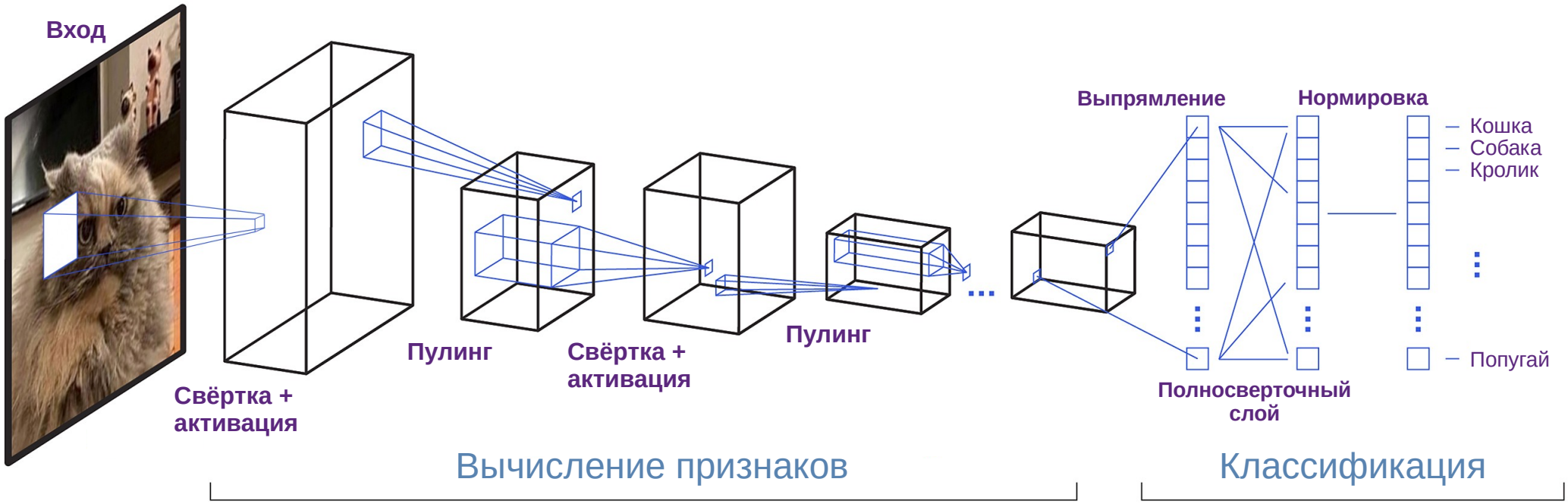
Нейронные сети применяются при решении задач в различных областях, например:

- Работа с изображениями и видеопоследовательностями
- Обработка естественного языка
- Распознавание речи

Настоящая работа относится к оптимизации вычислений нейронных сетей на уровне программной платформы

# Введение (2)

Пример свёрточной нейронной сети для классификации изображений




https://developer.qualcomm.com/blog/exploring-different-types-neural-networks

# Постановка задачи

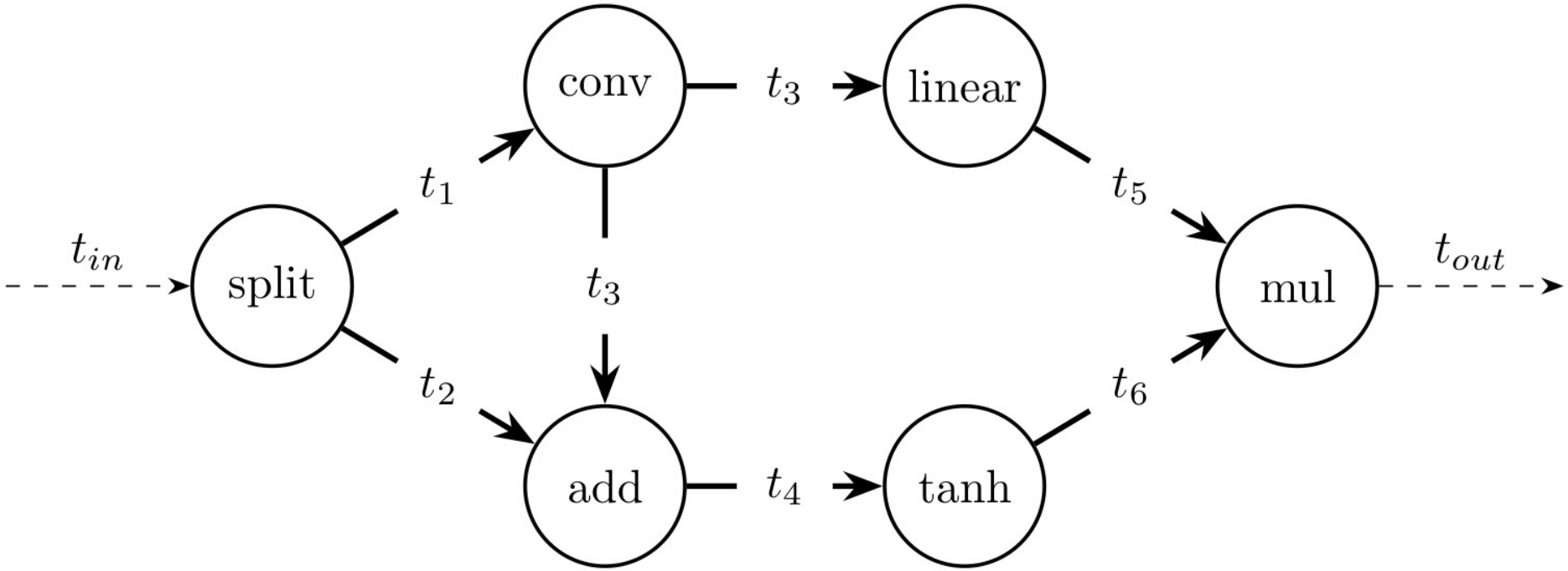
- Исследовать методы оптимизации используемой памяти и затраченного времени при вычислении предсказаний нейронных сетей
- Разработать метод одновременной оптимизации памяти и времени
- Реализовать и интегрировать метод в программно-аппаратную часть библиотеки искусственного интеллекта MindSpore
- Эмпирически исследовать разработанный метод, сравнить его с другими существующими подходами

# Библиотека MindSpore

Реализуемая программно-аппаратная часть библиотеки MindSpore использует прикладной программный интерфейс Vulkan, который предполагает низкоуровневую работу с памятью графического процессора

- Синхронизация вычислений осуществляется посредством размещения барьеров памяти в очереди команд

# Пример ациклического графа вычислений



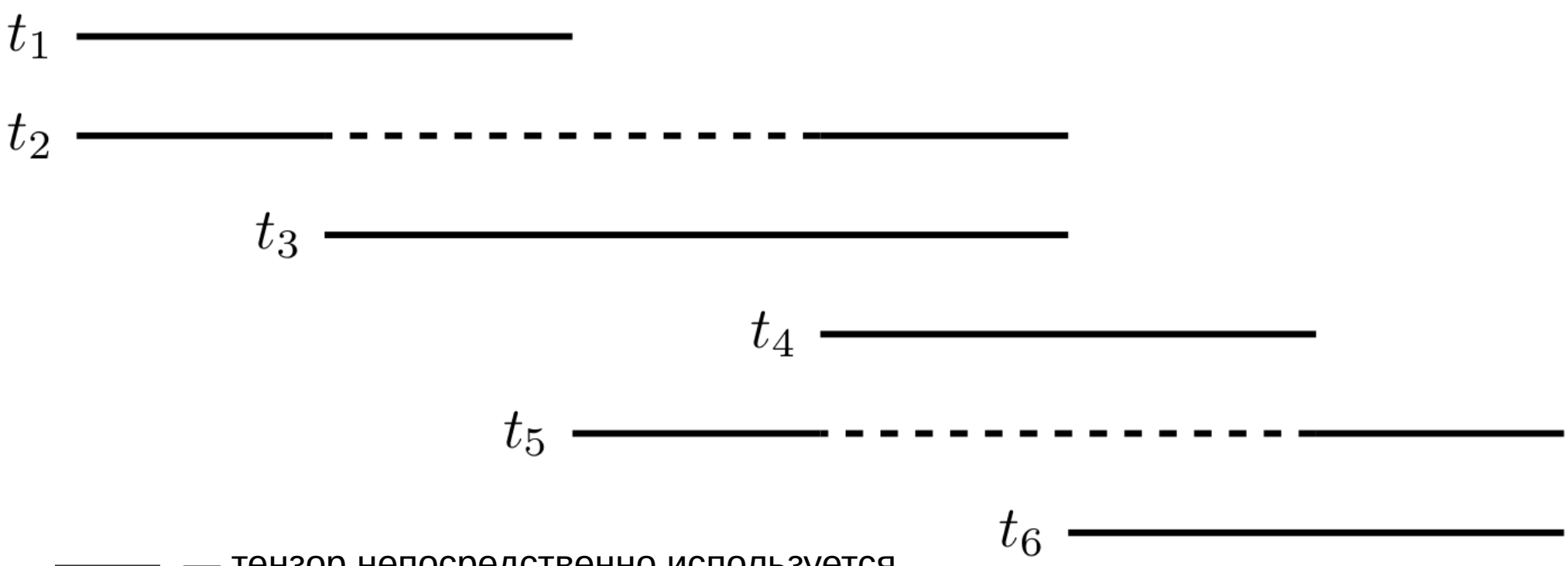

Вершины — операции

Рёбра — тензоры

# Время жизни тензоров

Одна из возможных топологических сортировок

| split | conv | linear | add | tanh | mul |
|---|---|---|---|---|---|




— тензор непосредственно используется

— тензор нужно хранить для дальнейших вычислений

# Задача оптимального распределения памяти (1)

- Хотим выделить для каждого тензора блок в памяти
- Минимизируем суммарный объем памяти при следующем ограничении:

  Тензоры с пересекающимися временами жизни не могут использовать одну и ту же память

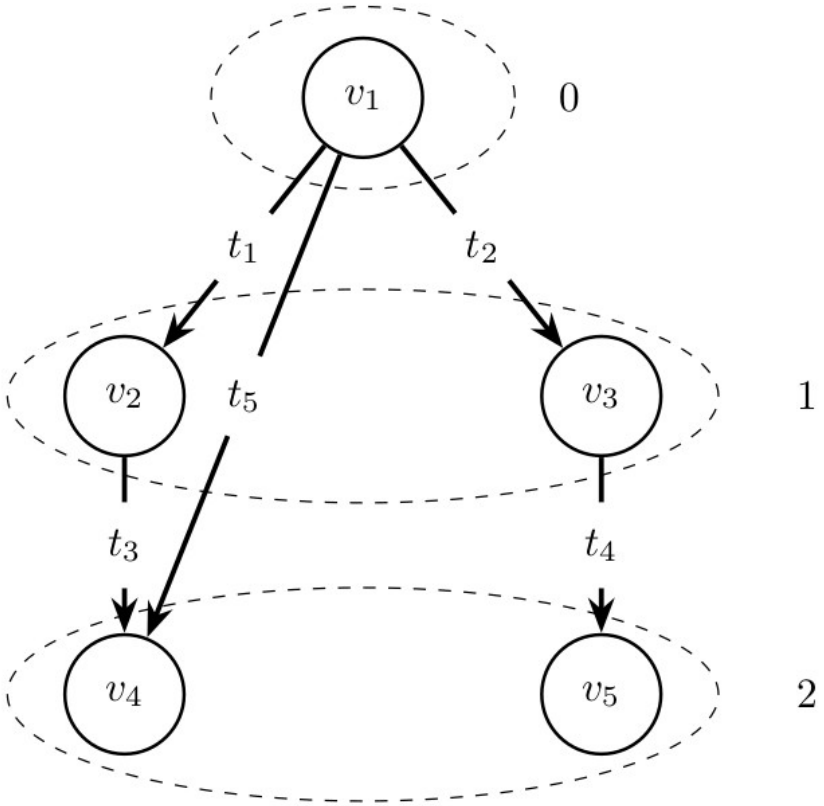
# Задача оптимального распределения памяти (2)

Данную задачу можно эффективно решить приближённо различными эвристическими алгоритмами, например: [1], [2].


1. Profile-guided memory optimization for deep neural networks / T. Sekiyama [и др.] // arXiv preprint arXiv:1804.10001. — 2018.

2. *Pisarchyk Y.*, *Lee J.* Efficient memory management for deep neural net inference // arXiv preprint arXiv:2001.03288. — 2020.

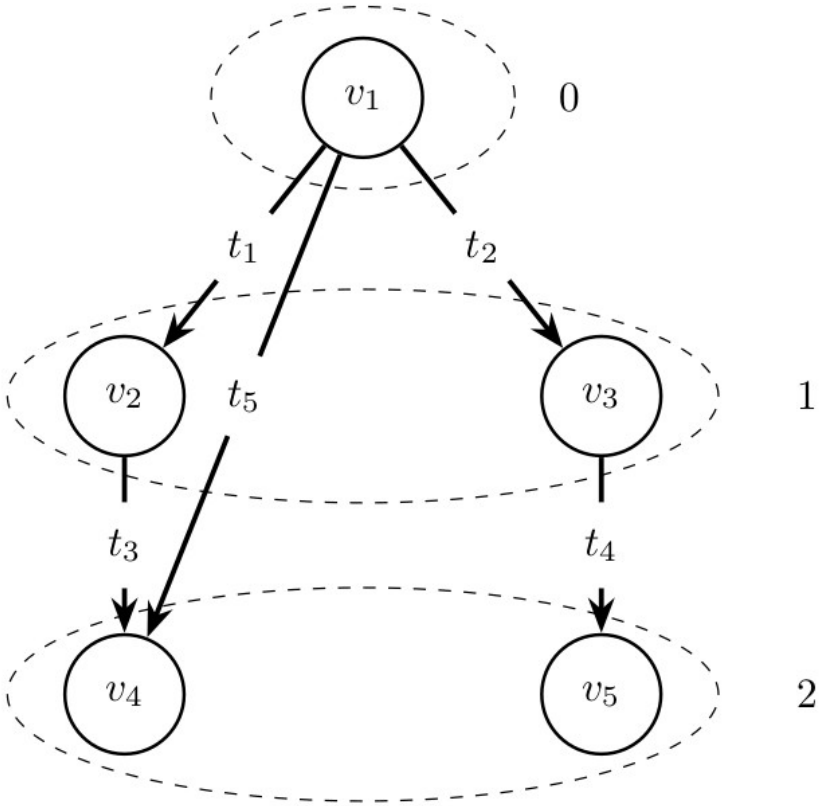
# Задача параллельных вычислений (1)




Пример ациклического графа вычислений, в котором возможно параллельное вычисление некоторых вершин

Пунктирными линиями обведены подмножества независящих друг от друга вершин

# Задача параллельных вычислений (2)




$L_i = \{v \mid maxdist(v) = i\}$ — слои [1]


1. Node-level parallelization for deep neural networks with conditional independent graph / F. Zhou [и др.] // Neurocomputing. — 2017. — июнь. — т. 267.

# Проблема

Подходы к решению рассмотренных задач конфликтуют между собой:

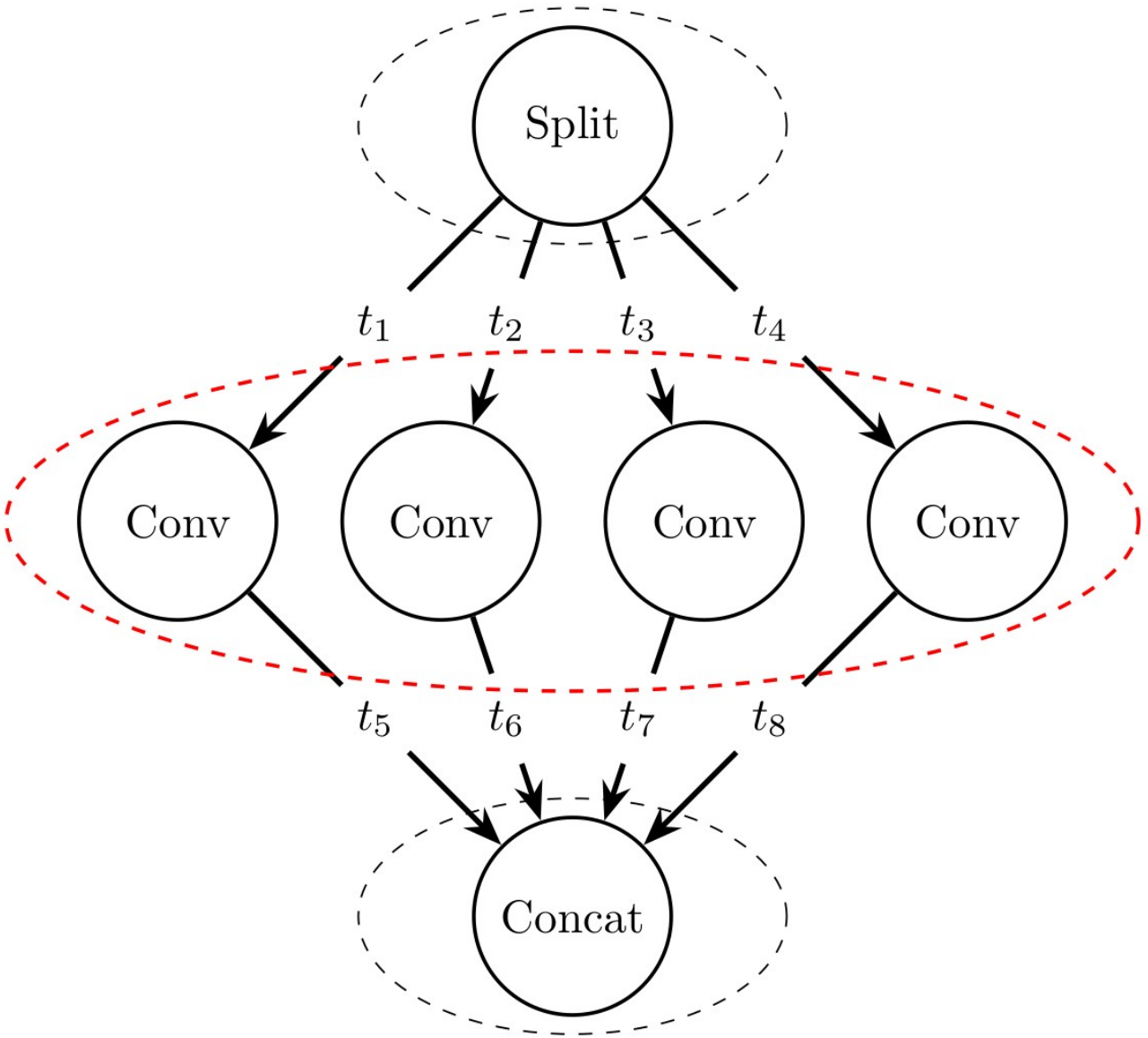
- параллелизация вычислений накладывает дополнительные ограничения на повторное использование памяти и может потребовать бóльший объем памяти для вычислений

# Предложенный метод (1)

1) Каждый слой разбивается на некоторое количество групп
2) Между вычислениями групп размещаются барьеры памяти
3) Вершины в группах вычисляются параллельно

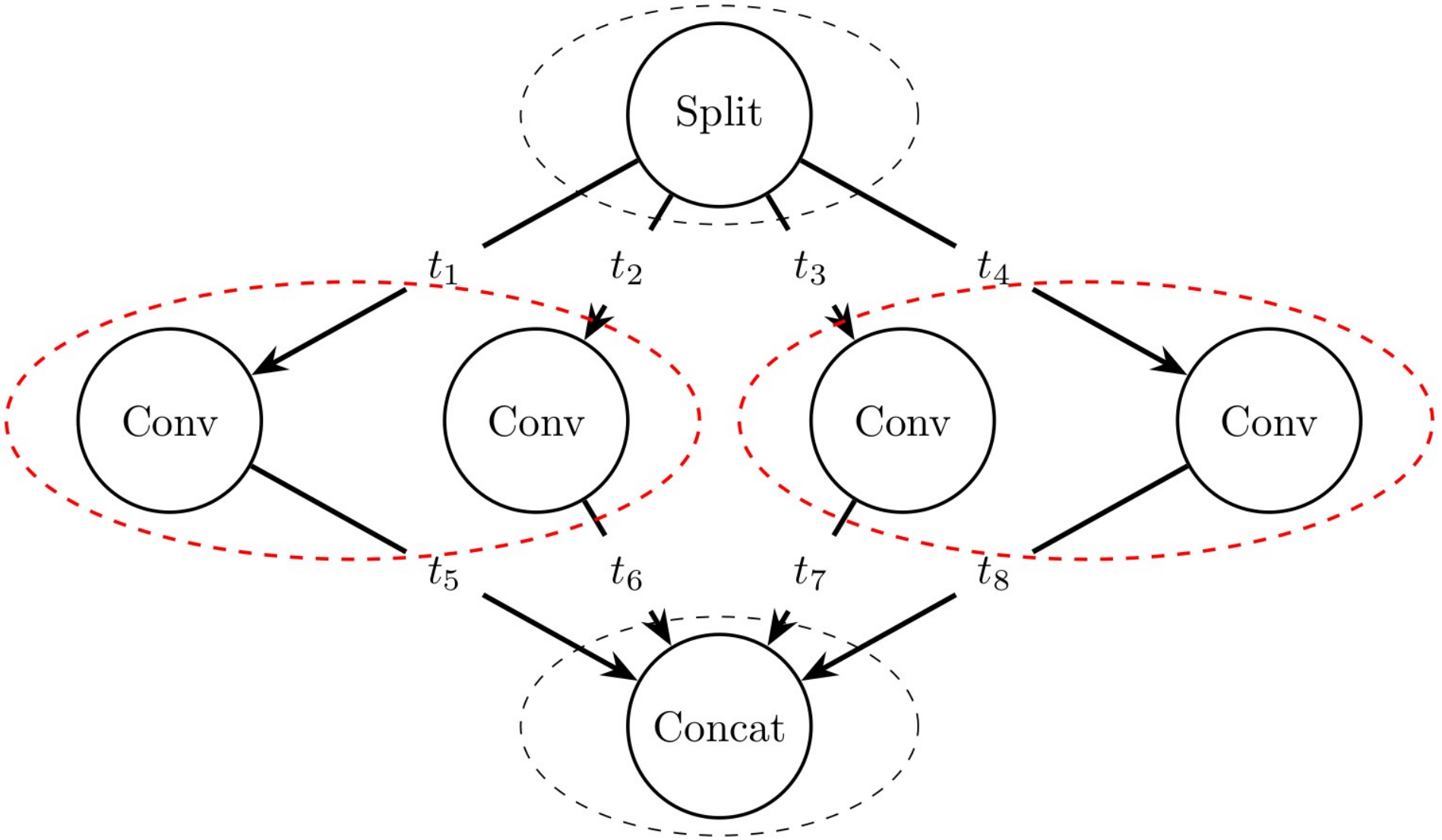
- Задействованные одной и той же группой тензоры не могут использовать общую память

# Предложенный метод (2)



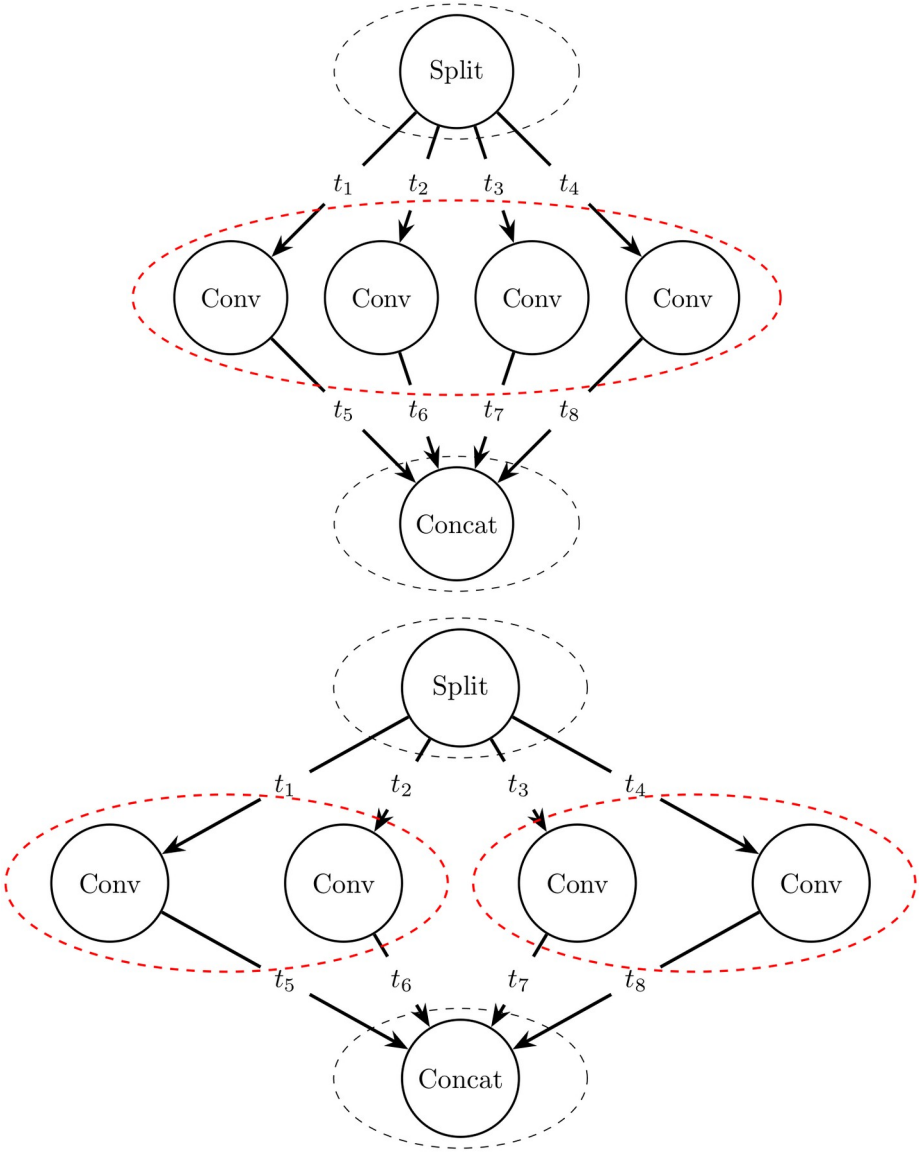

**1 группа**

# Предложенный метод (3)




**2 группы**

# Предложенный метод (4)



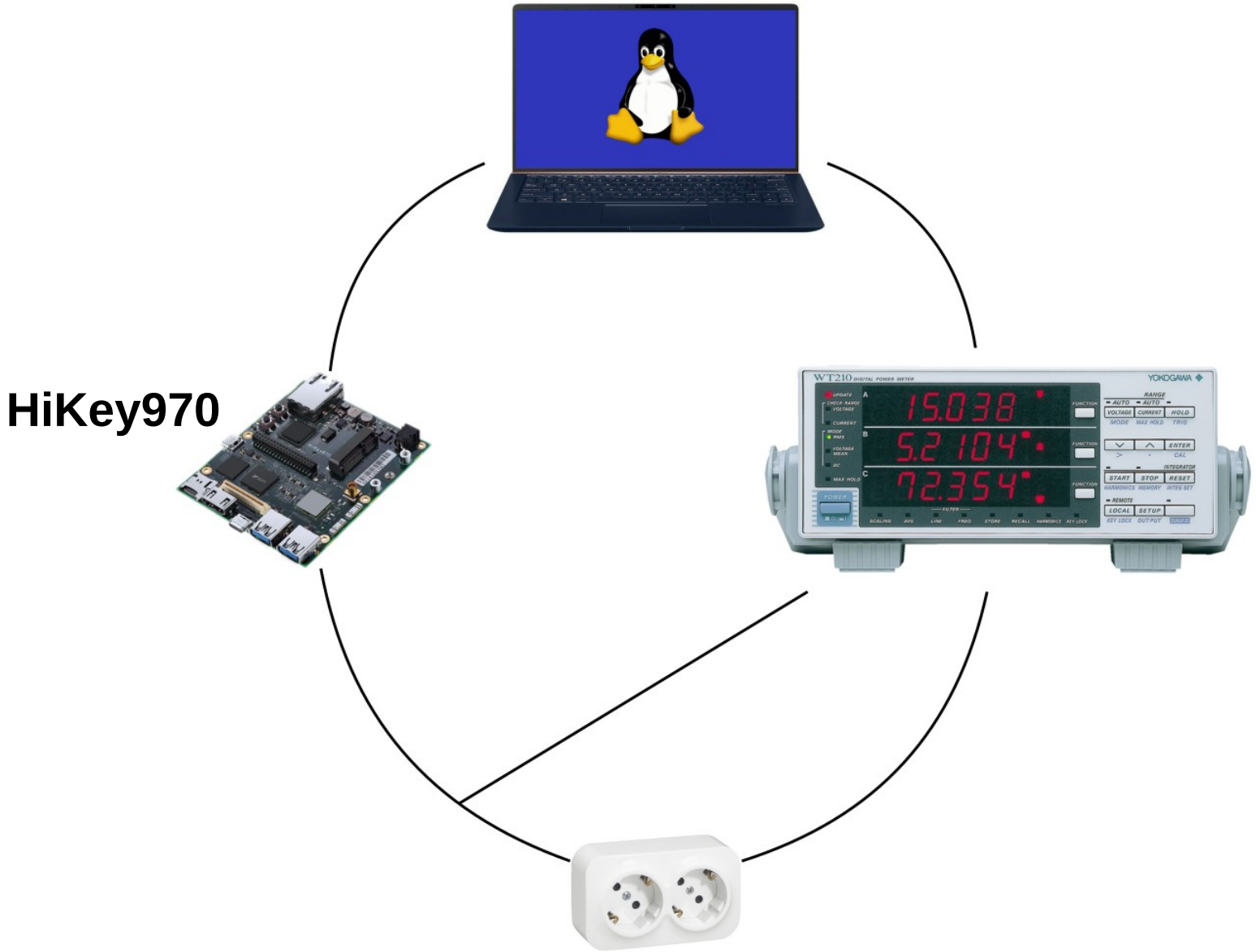

- **1 группа**

$$Mem = \max\{w(t_1) + w(t_2) + w(t_3) + w(t_4), \\ w(t_5) + w(t_6) + w(t_7) + w(t_8)\}$$

- **2 группы**

$$Mem = \max\{w(t_1) + w(t_2), \\ w(t_3) + w(t_4), \\ w(t_5) + w(t_6) + w(t_7) + w(t_8)\}$$

# Экспериментальная установка

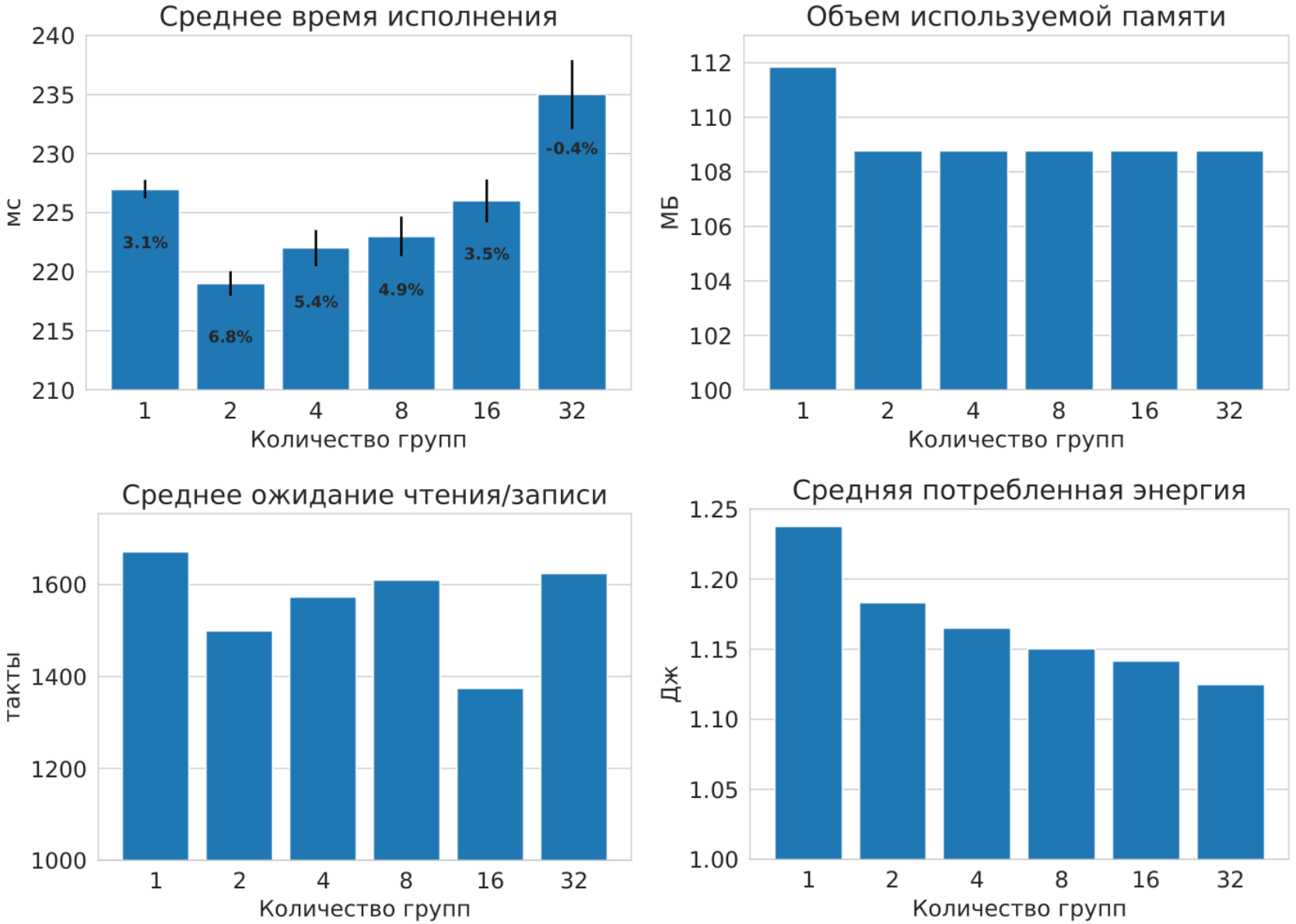


# Результаты (1)

| Метод | | ResNeXt | GoogLeNet | MobileNetV2 | Xception | InceptionV3 |
|---|---|---|---|---|---|---|
| Базовая реализация | время, мс | 234 ± 5 | 80 ± 3 | 81 ± 3 | 291 ± 2 | 369 ± 4 |
| | память, Мб | 393.7 | 67.9 | 78.3 | 268.5 | 198.1 |
| Минимизация памяти | время, мс | 235 ± 6 | 82 ± 1 | 85 ± 4 | 293 ± 2 | 370 ± 3 |
| | память, Мб | **108.8** | **42.4** | **30.3** | **113.2** | **122.6** |
| Послойное вычисление | время, мс | 228 ± 2 | **77 ± 1** | **80 ± 3** | **290 ± 2** | **361 ± 3** |
| | память, Мб | 393.7 | 67.9 | 78.3 | 268.5 | 198.1 |
| Мой метод (1 группа) | время, мс | 227 ± 2 | 78 ± 2 | 82 ± 3 | **290 ± 2** | **361 ± 4** |
| | память, Мб | 111.8 | **42.4** | **30.3** | **113.2** | **122.6** |
| Мой метод (2 группы) | время, мс | **219 ± 2** | 78 ± 1 | 83 ± 4 | 291 ± 2 | 364 ± 3 |
| | память, Мб | **108.8** | **42.4** | **30.3** | **113.2** | **122.6** |

Для результатов замеров времени приведено:
среднее значение по выборке ± два среднеквадратичных отклонения

# Результаты (2)




Результаты применения предложенного метода к сети ResNeXt

# Заключение (1)

- Рассмотрены существующие подходы к решению задач оптимизации времени и памяти
- Предложен метод, объединяющий эвристику оптимизации памяти с параллелизацией вычислений
- Метод реализован и протестирован в программно-аппаратной части библиотеки MindSpore с использованием Vulkan API

# Заключение (2)

- Проведено сравнение предложенного метода с другими на разных моделях нейронных сетей на мобильном устройстве
- Эмпирически показано, что организация обращений в память при различных конфигурациях метода может оказывать существенное влияние на время вычислений
- Для свёрточной нейронной сети ResNeXt получено ускорение на **6.85%** при экономии памяти на **72.4%**